REMARKS/ARGUMENTS

Amendments were made to the specification to correct errors and to clarify the specification. No new matter has been added by any of the amendments to the specification.

Claims 1-10 are pending in the present application. In this Amendment, Applicant has amended claims 1, 2, and 10. Applicant has canceled claims 11-20 from further consideration in this application. Applicant is not conceding that the subject matter encompassed by claims 1 and 11-20, prior to this amendment are not patentable over the art cited by the Examiner. Claim 1 was amended and claims 11-20 were canceled in this amendment solely to facilitate expeditious prosecution of the remaining claims. Applicants respectfully reserve the right to pursue claims, including the subject matter encompassed by claims 1, and 11-20, as presented prior to this amendment and additional claims in one or more continuing applications. Reconsideration of the claims is respectfully requested.

I. 35 U.S.C. § 103, Obviousness

The Examiner has rejected claims 1-20 under 35 U.S.C. § 103 as being unpatentable over *Hitz et al.,* File Access Control in a Multi-Protocol File Server, U.S. Patent No. 6,457,130, September 24, 2002 (hereinafter "*Hitz*") in view of *Montague et al.,* Controlling Access to Objects on Multiple Operating Systems, U.S. Patent No. 5,761,669, June 2, 1998 (hereinafter "*Montague*"). This rejection is respectfully traversed.

Regarding the rejection of claim 1, the Examiner states:

> Regarding Claims 1 and 20, Hitz discloses a method for managing access control lists in a filesystem (Figure I), the method comprising:
>
> associating two or more access control lists (first access control model and second access control model) with a given filesystem object, (Figure 1, element 1 12) in a heterogeneous filesystem (e.g., system 100, col. 1, lines 40-65), wherein the heterogeneous filesystem (Figure 1, element 100) comprises two or more differing types of filesystems (e.g., UNIX and NT) (Figure 1) (col. 4, lines 7-11);
>
> responsive to receiving, from a requestor (Figure 1, element 120), a request (Figure 1, element 121) for an access control list associated with the given filesystem object (Figure I, element 112) (col. 3, line 62 - col. 4, lines 11), determining a filesystem type of the requester (col. 3, lines 44-51 and col. 5, lines 35-67).
>
> Hitz does not disclose "returning an access control list from the two or more access control lists for the given filesystem object matching the filesystem type of the requestor."

However, Montague expressly discloses returning an access control list from the two or more access control list for the given filesystem object matching the filesystem type of the requestor (Figures 2 and 11, col. 3, lines 17-25 and col. 15, lines 10-17).

Therefore, it would have been obvious at the time the invention was made to a person having ordinary skill in the art to have incorporated Montague's invention within Hitz to include returning an access control list from the two or more access control lists for the given filesystem object matching the filesystem type of the requestor. One of ordinary skill in the art would have been motivated to do this because it would provide controlling access to objects on a network (Montague, col. 1, lines 14-15).

Office Action dated November 26, 2007, pp. 3-4.

The Examiner bears the burden of establishing a *prima facie* case of obviousness based on prior art when rejecting claims under 35 U.S.C. § 103. *In re Fritch*, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). The prior art reference (or references when combined) must teach or suggest all the claim limitations. *In re Royka,* 490 F.2d 981, 180 USPQ 580 (CCPA 1974). In determining obviousness, the scope and content of the prior art are...determined; differences between the prior art and the claims at issue are...ascertained; and the level of ordinary skill in the pertinent art resolved. Against this background the obviousness or non-obviousness of the subject matter is determined. *Graham v. John Deere Co.*, 383 U.S. 1 (1966). Often, it will be necessary for a court to look to interrelated teachings of multiple patents; the effects of demands known to the design community or present in the marketplace; and the background knowledge possessed by a person having ordinary skill in the art, all in order to determine whether there was an apparent reason to combine the known elements in the fashion claimed by the patent at issue. *KSR Int'l. Co. v. Teleflex, Inc.*, No. 04-1350 (U.S. Apr. 30, 2007). Rejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness. *Id.* (citing *In re Kahn*, 441 F.3d 977, 988 (CA Fed. 2006)).

Independent claim 1 recites:

A method for managing access control lists in a filesystem, the method comprising:

associating two or more access control lists with a given filesystem object in a heterogeneous filesystem, wherein the heterogeneous filesystem comprises a native filesystem and one or more additional differing filesystems;

responsive to receiving, from a requestor, a request for an access control list associated with the given filesystem object, determining a filesystem type of the requestor; and

responsive to a determination that an access control list in the two or more access control lists associated with the given filesystem object matches the filesystem type of the requestor, returning an access control list from the two or more access control lists for the given filesystem object matching the filesystem type of the requestor.

I.A. Associating one or more access control lists with a given file system object

The proposed combination of references, considered as a whole, does not teach or suggest the features of "associating two or more access control lists with a given filesystem object in a heterogeneous filesystem, wherein the heterogeneous filesystem comprises a native filesystem and one or more additional differing filesystems," as in claim 1. The Examiner cites to *Hitz* at column 1, lines 40-65 which states:

> For example, a first access control model in common use is associated with the Unix operating system (or a variant thereof). This first access control model associates permissions with each file for a file owner, an owner's group, and all other users. These permissions allow access (for the owner, group, or all other users) to read, write, or execute the indicated file. This first access control model is typically implemented by the NFS ("Network File System") file server protocol, possibly augmented with an adjunct file-locking protocol, NLM ("Network Lock Manager"). A second access control model in common use is associated with the Windows NT operating system. This second access control model associates an ACL (access control list) with each file, each entry in the ACL specifying an individual user, a group of users, or all users. Each entry can allow access (for the specified users) to read, write, or execute the indicated file, or can specifically deny access. This second access control model is typically implemented by the CIFS ("Common Internet File System") protocol. However, NT devices can also use the NFS protocol by means of the "PC NFS" implementation, and Unix devices can also manipulate POSIX ACLs. These two access control models in common use differ in significant ways, including (1) what permissions can be assigned to a file, (2) with what granularity of specificity permissions can be assigned, and (3) how users are identified so as to match them with permissions.

Here, *Hitz* describes two types of access control models. One type of access control model in common use is associated with the Unix operating system and the other type of access control model is associated with Windows NT. *Hitz* does not teach or suggest that these two types of access control models are associated with a given file system object.

The Examiner also cites to figure 1 of *Hitz* which illustrates as follows:

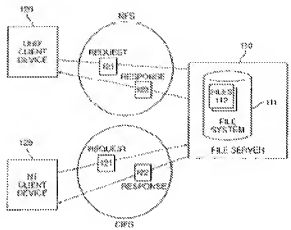

Figure 1 shows a system that includes a file server and a set of files on a file system. Figure 1 does not show any access control lists associated with a file or other file system object in the file server. Moreover, *Hitz* only shows a single file system, filesystem 111, on the system. *Hitz* does not teach or suggest a heterogeneous filesystem having a native file system and one or more additional differing filesystems in this figure or in any other section of the cited reference.

For example, one portion of *Hitz* describing Figure 1 states:

> The file server 110 is disposed for receiving file server requests 121 from the client devices 120. The file server 110 parses each request 121, determines whether the operation requested in the request 121 is allowed (for the client device 120 that sent the request 121 and for the one or more target files 112 specified by the request 121). If allowed, the file server 110 performs that operation on the one or more target files 112.

Hitz, column 3, lines 44-51.

Here, *Hitz* states that the file server in figure 1 receives requests from client devices. The file server determines whether the requested operation is allowed for the client device that sent the request and for the target files specified by the request. If the operation is allowed, the file server performs that operation. Although *Hitz* describes receiving requests from multiple client devices, *Hitz* does not teach or suggest two or more access control lists associated with a given filesystem object in a heterogeneous filesystem having two or more differing types of filesystems.

The Examiner also cites to *Hitz* at column 4, lines 7-11 which states:

> The file server 110 supports more than one access control model, including a "Unix Perms" access control model (herein "Unix security style") and an "NT ACL" access control model (herein "NT security style").

Again, this portion of *Hitz* states that the file server supports more than one access control model. Although the file server supports multiple access control models, *Hitz* does not teach or suggest associating multiple access control models with a single file system object. In fact, *Hitz* explicitly states that a given file in the file system has one designated access control model out of a plurality of access control model supported by the file server in column 2, lines 36-47 which states:

> The invention provides a method and system for enforcing file access control among client devices using multiple diverse access control models and multiple diverse file server protocols. A multi-protocol file server identifies each file with one particular access control model out of a plurality of possible models, and enforces that one particular model for all accesses to that file. When the file server receives a file server request for that file using a different access control model, the file server translates the access control limits for that file into no-less-restrictive limits in the different model. The file server restricts access by the client device using the translated access control limits.

As can be seen, *Hitz* expressly teaches that each file is associated with one particular access control model out of a plurality of possible models. When a request is received using a difference access control model, the file server translates the access control limits for the file into the different model. Thus, *Hitz* fails to teach associating two or more access control lists with a given filesystem object in a heterogeneous filesystem.

I.B. Returning an access control list from the two or more access control lists

The proposed combination of references, considered as a whole, does not teach or suggest the feature of "responsive to a determination that an access control list in the two or more access control lists associated with the given filesystem object matches the filesystem type of the requestor, returning an access control list from the two or more access control lists for the given filesystem object matching the filesystem type of the requestor", as claimed in claim 1. The Examiner concedes that *Hitz* does not teach returning an access control list from the two or more access control lists for the given filesystem object matching the filesystem type of the requestor.

However, the Examiner believes *Montague* discloses this feature in Figure 2 which illustrates:

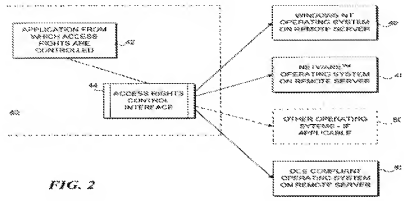

FIG. 2

Here, *Montague* shows a workstation having an access rights control interface on the workstation. The remote servers use operating systems, such as Windows NT and other operating systems. Although *Montague* shows remote servers using different operating systems and a workstation having an access rights control interface, *Montague* still fails to disclose or suggest returning an access control list from two or more access control lists associated with a given filesystem object. In fact, *Montague* does not show a filesystem object associated with two or more access control lists or returning an access control list from two or more access control lists associated with a given filesystem object matching the filesystem type of the requestor in this figure or in any other section of the reference.

The Examiner also cites to *Montague* at column 3, lines 17-25 which states:

> In response to a request by the user, the network operating system determines the trustees that can have the specific access rights assigned to them and returns a list of the trustees in a format that is independent of the network operating system on which the specific access rights are to be set. A user must have the right to grant access to the entity and can only affect the access rights of a trustee on the list. Another step of the method is to enable a user to view a trustee's access permissions to an entity.

This portion of *Montague* describes an operating system returning a list of trustees in a format that is independent of the operating system on which the access rights are to be set. Returning a list of trustees does not teach or suggest returning an access control lists associated with a filesystem object from two or more access control lists associated with the filesystem object.

Instead, *Montague* is merely returning a list of groups or individuals that can have access rights assigned to them. *See Montague* at column 9, line 57.

The Examiner cites to *Montague* figure 11 which illustrates as follows:

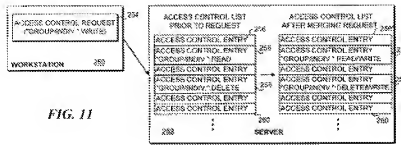

FIG. 11

Figure 11 shows an access control list that is modified by an access control request to add a write permission to a specific group or individual. A portion of *Montague* describing figure 11 states:

> FIG. 11 illustrates the relationship between an access control request 254 that is created by a trustee at a workstation 250 to modify the permissions for an entity, producing an appropriate ACE in an ACL 256, which is maintained on a server 252. In the example shown in this FIG., access control request 254 is a request to grant Write permissions to a specific group/individual, for a particular entity, such as a file. To merge this access control request with the existing ACEs maintained in ACL 256, the operating system searches the ACL to identify account IDs matching the specific group/individual account ID corresponding to the name of the individual/group included in the access control request. In this example, two ACEs are found with the same account ID, the first granting Read permissions to the entity and the second granting Delete permissions to the entity. Since the entity is simply a file, and not a container, inheritance attributes are not involved in merging the access control request with the ACL. As a result, as shown in ACL 256, the access control request is merged into the ACL, producing ACEs 258, the first of which grants Read/Write permissions to the account ID for the specific group/individual in the request, and the second granting Delete/Write permissions to the file for the specific group/individual specified by the account ID. Any other ACEs 260 on the entity are not affected by the grant of the request.

As shown above, figure 11 illustrates a creating an access control entry in an access control list adding a write permission for a group or an individual for a particular entity, such as a file. Figure 11 only shows a single access control list for the entity before the new access control entry is added and after the new access control entry is added granting read and writes permission for the group or individual. Figure 11 does not show multiple access control lists associated with the file

or returning an access control list from the multiple access control lists matching the filesystem type of the requestor.

Finally, the Examiner cites to *Montague* at column 15, lines 10-17, which discloses:

> The technique employed under the WINDOWS NT operating system for handling an access control request recognizes that ACEs in the ACL can be inter-related. For example, different ACEs in the ACL may grant permissions to different entities, for the same account ID. Under the present invention, a trustee or an application can readily formulate an access request without having any knowledge of the related ACEs already entered in the ACL.

In this section, *Montague* discloses that access control entries in a single access control lists can be interrelated. A user can formulate an access request without having knowledge of the related entries in an access control list. This portion of *Montague* again fails to disclose two or more access control lists associated with a given filesystem object and returning an access control list from the two or more access control lists that matches the filesystem type of the requestor. Moreover, *Montague* fails to disclose a heterogeneous filesystem in which the two or more access control lists are associated with a given filesystem object. Therefore, *Montague* fails to make up for the deficiencies of *Hitz*.

Therefore, neither *Hitz* nor *Montague* teach or suggest responsive to a determination that an access control list in the two or more access control lists associated with the given filesystem object matches the filesystem type of the requestor, returning an access control list from the two or more access control lists for the given filesystem object matching the filesystem type of the requestor", as claimed in claim 1. For this reason, the proposed combination of cited references, considered as a whole, also does not teach or suggest this claimed feature. Hence, under the standards of *In re Royka*, the Examiner failed to state a *prima facie* obviousness rejection against claim 1.

I.C. Dependent claims 2-10

Claims 2-10 depend from claim 1. Therefore, claims 2-10 are allowable over *Hitz* and *Montague* for at least the reasons set forth above with regard to claim 1. In addition, claims 2-10 recite additional combinations of features that are not taught or suggested by the combination of the cited references. For example, claim 2 recites "determining whether an access control list in the two or more access control lists associated with the given filesystem object matches the filesystem type of the requestor, wherein the two or more access control lists associated with the

filesystem object comprises multiple access control list formats". As discussed above, *Hitz* and *Montague* both fail to teach or suggest associating two or more access control lists with a given filesystem object. Moreover, the cited references, when taken as a whole, fail to teach or suggest the two or4 more access control lists comprises multiple access control list formats. *Hitz* at column 5, lines 11-22 states:

> Although a preferred embodiment of the invention is described with regard to Unix security style and NT security style, the invention can readily be used with other access control models, such as the "POSIX ACL" access control model supported by some Unix devices, and by some other operating systems. The concepts and features of the invention described herein can readily be used in a file server 110 supporting the "POSIX ACL" access control model in addition to or instead of the access control models described in detail herein, without further invention or undue experimentation. Accordingly, the scope and spirit of the invention includes such file servers and methods for their use.

This portion of the reference states that the system of *Hitz* may be used with access control models supported by other operating systems, such as POSIX ACL access control model. Again, *Hitz* only discloses a filesystem that supports the POSIX ACL access control model in addition to or instead of the Unix security style and NT security style access control models. *Hitz* does not teach or suggest associated two or more different access control lists with a single filesystem object. Thus, the cited references fails to teach or suggest the features recited in amended claim 2.

II. *Hitz* teaches away from the presently claimed invention in claim 1.

As discussed above, *Hitz* teaches identifying each file with one particular access control model out of a plurality of access control models rather than associating two or more access control lists with a given filesystem object. *Hitz* expressly teaches away from associating multiple different access control lists with a single filesystem object, such as, without limitation, a file. In contradistinction, claim 1 claims "associating two or more access control lists with a given filesystem object in a heterogeneous filesystem, wherein the heterogeneous filesystem comprises a native filesystem and one or more additional differing filesystems." Thus, one of ordinary skill in the art would not find it obvious to modify *Hitz* to reach the presently claimed invention in claim 1.

Therefore, the rejection of claims 1-20 under 35 U.S.C. § 103 has been overcome.

III. The Examiner Fails to Present a *Prima facie* Case of Obviousness Because the Examiner Has Not Stated a Proper Reason to Combine the References.

Additionally, the Examiner failed to state a *prima facie* obviousness rejection against claim 1 because the Examiner failed to state a proper reason to combine the references under the standards of *KSR Int'l*. As shown above, *Hitz* and *Montague* simply do not teach or suggest what the Examiner believes these references to teach and suggest. Therefore, due to the large differences between the cited references and the features in claim 1, the reasoning provided by the Examiner to combine the references rests on inherently flawed reasoning. For this reason, the Examiner did not state a proper, rational reason to combine the references as required by *KSR Int'l*.

IV. Conclusion

It is respectfully urged that the subject application is patentable over the cited references and is now in condition for allowance.

The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: February 20, 2008

Respectfully submitted,

/Mari A. Stewart/

Mari A. Stewart
Reg. No. 50,359
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 385-8777
Attorney for Applicants